УДК: 004.89 + 623.4 https://doi.org/10.15407/jai2024.04.173

Ю. В. Мельник[1], Є. Д. Лукацький[2]
[1]Державний університет інформаційно-комунікаційних технологій, Україна
вул. Солом'янська, 7, Київ 03110
[2]Національний авіаційний університет, Україна
пр. Гузара Любомира, 1, Київ 03058
[1]melnik_yur@ukr.net
[2]evgeniy.lukatsky@gmail.com
[1]https://orcid.org/0000-0002-5028-8749
[2]https://orcid.org/0000-0003-3352-3230

АЛГОРИТМИ ПРИВ'ЯЗКИ МІСЦЕЗНАХОДЖЕННЯ БПЛА ЗІ ЗБІГОМ СЦЕНИ

Yu. Melnyk[1], Ye. Lukatskyi[2]
[1]State University of Information and Communication Technologies, Ukraine
Solomianska St., 7, Kyiv 03110
[2]National Aviation University, Ukraine
Lubomira Guzara av., 1, Kyiv 03058
[1]melnik_yur@ukr.net
[2]evgeniy.lukatsky@gmail.com
[1]https://orcid.org/0000-0002-5028-8749
[2]https://orcid.org/0000-0003-3352-3230

ALGORITHMS FOR UAV LOCALIZATION WITH SCENE MATCHING

Анотація. Стаття присвячена проблемі навігації безпілотних літальних апаратів та використанню можливостей штучного інтелекту для вирішення існуючих проблем. Здійснено огляд історії використання систем керування балістичними та крилатими ракетами, починаючи з 40-х рр. ХХ ст. Визначено недоліки існуючих систем керування за допомогою інерційних систем, проблеми, що виникають в умовах воєнних конфліктів. Встановлено основні етапи процесу навігації, зокрема шляхом порівняння зображень. Здійснено аналіз існуючих методів порівняння зображень для цілей навігації з використанням різних систем, визначено їх переваги та недоліки, проаналізовано математичні моделі цих процесів. Запропоновано шляхи вирішення проблем існуючих методів керування безпілотними літальними апаратами, зокрема шляхом використання методу ковзного вікна з метою зменшення необхідних ресурсів та скорочення часу прийняття навігаційних рішень у процесі польоту у режимі реального часу із залученням можливостей штучного інтелекту.
Ключові слова: безпілотні літальні апарати, навігація, системи керування, інерційна система навігації, датасет, навчання системи, штучний інтелект.

Abstract. This article addresses the issue of unmanned aerial vehicle (UAV) navigation and the application of artificial intelligence (AI) to overcome current challenges. It reviews the historical use of control systems for ballistic and cruise missiles dating back to the 1940s. The shortcomings of existing control systems that rely on inertial systems and the challenges faced in aerial conflict scenarios are identified. The main stages of the navigation process, particularly through image comparison, are outlined. An analysis of existing image comparison methods for navigation purposes using various systems is presented, detailing their advantages and disadvantages and examining the mathematical models underlying these processes. The paper proposes solutions to the challenges faced by existing UAV control methods, specifically by utilizing a sliding window approach to reduce required resources and shorten decision-making time for real-time flight navigation through AI integration.
Keywords: Unmanned aerial vehicles, navigation, control systems, inertial navigation system, dataset, system training, artificial intelligence.

Питання навігації та визначення місцезнаходження безпілотних літальних апаратів (БПЛА) є одним з найважливіших для успішного виконання завдань у сучасних складних умовах радіо-електронної протидії. Зазвичай, після появи систем супутникової навігації та мереж GNSS, безпілотні літальні апарати у навігації покладалися на інформацію від бортових приймачів супутникової навігації, що дозволяло їм з високою точністю визначати своє місцезнаходження в умовах цивільного використання у мирний час.

Однак, під час військових дій сторони збройних конфліктів використовують різноманітні засоби протидії РЕБ, зокрема глушіння або підміну даних супутникової навігації, вплив на радіоканали та інші методи протидії, що негативно позначаються на можливості використання БПЛА як у військовій, так і у цивільній сферах.

З іншого боку, починаючи з 40-х років XX століття, перед дослідниками стояли завдання керування балістичними та крилатими ракетами для забезпечення керованості та підвищення точності нанесення ударів вглиб території супротивника. Першим автономним літальним апаратом, що використовував інерційну систему навігації, була німецька балістична ракета V-2, яка здійснила перший політ у 1942 році [1].

З того часу розвиток систем управління крилатих, балістичних ракет та інших безпілотних літальних апаратів пройшов шлях від інерційних систем керування до сучасних систем на основі глобальних навігаційних супутникових систем. У перші роки розвитку ракетних технологій основною технологією керування була інерційна система, яка працювала на основі гіроскопів та акселерометрів. Ця технологія дозволяла визначати положення ракети у просторі без використання зовнішніх орієнтирів. Однак точність таких систем знижувалася з часом протягом польоту через накопичення похибок.

У 1970-ті роки було упроваджено систему DSMAC (Digital Scene-Matching Area Correlator), яка дозволила поліпшити точність завдяки використанню цифрових зображень місцевості для порівняння з еталонними картами. Ця технологія активно застосовувалась у крилатих ракетах, таких як BGM-109 Tomahawk, для наведення на ціль у складних умовах, де використання лише інерційної навігації не забезпечувало достатньої точності. Важливим елементом було також використання системи огинання рельєфу місцевості (TERCOM). Це дозволяло ракетам використовувати дані про рельєф для здійснення польоту на низьких та наднизьких висотах з метою запобігання виявлення радарами ППО та ПРО [2]. У Радянському Союзі також активно розвивалися системи керування ракетами. Радянські вчені працювали над створенням аналогічних технологій для підвищення точності ракетних ударів. Наприклад, система управління для ракет Р-7 та наступних моделей включала інерційні елементи, а також було розроблено технології, подібні до DSMAC, з використанням обробки зображень для корекції траєкторії. Система огинання рельєфу місцевості (TERCOM) була вперше реалізована на крилатих ракетах Х-55, що були розроблені в 1980-х роках.

У перших версіях DSMAC використовувалася технологія кореляції на основі двовимірних зображень для оновлення навігації. Зображення місцевості порівнювалися з попередньо збереженими еталонними картами. Алгоритм DSMAC використовував порівняння бінарних зображень для визначення найкращого співпадіння між зібраними під час польоту кадрами і еталонними картами. Для обробки зображень використовувався алгоритм бінарної кореляції, який визначав співпадіння на основі кількості точок, що відповідають еталонному зображенню.

Для корегування траєкторії польоту DSMAC використовував спалах для освітлення сцени та знімав послідовність кадрів, які порівнювалися з еталонною картою. Щоб підтвердити правильність збігу і зменшити ймовірність помилкового співпадіння, використовувалася стратегія голосування з трьох кадрів.

Алгоритм TERCOM використовував висотоміри для порівняння рельєфу під ракетою з попередньо збереженими картами висот. Основний підхід полягав у зіставленні рельєфу для визначення місцезнаходження ракети відносно збереженої карти. Алгоритм використовував дані про вертикальні контури місцевості, щоб оновлювати позицію ракети. TERCOM намагався знайти найкращий збіг між виміряним профілем висоти та збереженою картою, використовуючи прості метрики для оцінки схожості.

Для покращення точності TERCOM обирав ділянки місцевості з унікальними характеристиками рельєфу, які забезпечували надійне оновлення позиції. Алгоритм також використовував дані місцевості, щоб зменшити вплив помилок, пов'язаних з погодними або сезонними змінами.

Основними відмінностями цих методів від сучасних були обмежені ресурси для обчислення та спрощені підходи до кореляції. У ранніх версіях DSMAC і TERCOM через обмеженість обчислювальних потужностей того часу використовували простіші обчислювальні методи, які не мали можливості виконувати складну обробку зображень або використовувати сучасні методи, такі як SIFT чи SURF. Кореляційний аналіз здійснювався на основі бінарних зображень з простими метриками для оцінки співпадіння, що не забезпечувало інваріантності до масштабу або повороту, як це є у сучасних алгоритмах.

Однак інтенсивний розвиток супутникової навігації призвів до того, що у сучасних системах керування здебільшого використовують GNSS. Це забезпечує високу точність завдяки супутниковим даним, однак досвід українсько-російської війни і війни «Сталеві мечі» в Ізраїлі, що тривають нині, демонструє, що інтеграція з інерційними системами залишається важливою для підвищення надійності та безперервності навігації в умовах, коли сигнал GNSS може бути заглушений або недоступний. Інтеграція DSMAC та інерційних систем також стала більш важливою та набула нових можливостей завдяки розвитку технологій обробки зображень та розпізнавання об'єктів, що дозволяє використовувати різні підходи для підвищення точності навігації.

Метою статті є визначення алгоритмів, які можуть застосовуватися для покрашення точності навігації за рахунок використання комбінованої системи керування з використанням підходу Terrain-aided Navigation (TAN), що дозволяє відмовитися від використання систем супутникової навігації.

Процес порівняння зображень можна розділити на два етапи:

1) попередня підготовка цифрових моделей місцевості, ортофотопланів та мап висот у зоні майбутнього польоту;
2) обробка вхідного відеопотоку на борту БПЛА та порівняння отриманих результатів з попередньо підготовленою базою ознак для визначення місцезнаходження.

Обидва етапи мають спільні елементи (методи визначення ознак, використання однакових алгоритмів обробки тощо), але водночас наявні і суттєві відмінності (доступні обчислювальні ресурси, об’єм вхідних даних, вимоги до реального часу).

Для обробки зображень зазвичай використовуються методи, засновані на виділенні ознак, такі як Scale-Invariant Feature Transform (SIFT) [3], Speeded-Up Robust Features (SURF) [4] та Oriented FAST and Rotated BRIEF (ORB) [5]. Ці методи виділяють ключові точки, які є інваріантними до масштабу, обертання та зміни освітлення. Відповідність цих ознак між зображеннями дозволяє оцінити геометричні перетворення і, відповідно, визначити відносне положення БПЛА. Також, на наш погляд, важливим варіантом визначення місцезнаходження є розпізнавання за рахунок машинного зору доріг, башт та інших лінійних та точкових орієнтирів, що є визначальними елементами місцевості. Це може допомогти зменшити необхідну підготовку через використання вже готових цифрових мап та методу тріангуляції.

Розглянемо основні методи визначення ознак, які активно використовуються зараз та містяться у бібліотеці OpenCV2.

Scale-Invariant Feature Transform (SIFT)

SIFT був розроблений Д. Лоу у 1999 році і був одним з перших алгоритмів для виділення ознак, що здобув значну популярність завдяки своїй стійкості до змін масштабу, обертання та освітлення. Він широко застосовується у комп'ютерному баченні для завдань розпізнавання об'єктів, мозаїки зображень і

навігації роботів. Алгоритм складається з кількох етапів:

1. Побудова зображень октави.

Алгоритм починається з побудови так званої "октави" зображення, яка складається з послідовності згладжених і зменшених версій початкового зображення. Для цього використовуються гаусові фільтри з різним масштабом.

Гаусова функція має вигляд:

$$G(x,y,\sigma) = \frac{1}{2\pi\sigma^2} e^{\frac{-x^2+y^2}{2\sigma^2}},$$

де σ визначає ширину гаусового фільтру і, відповідно, ступінь згладження.

2. Різниця гаусіанів (Difference of Gaussian, DoG).

SIFT використовує різницю гаусіанів (DoG), яка допомагає виділити області з високою частотою змін, тобто крапки і контури. Це робиться шляхом віднімання двох сусідніх зображень з різними значеннями σ:

$$D(x,y,\sigma) = G(x,y,k\sigma) - G(x,y,\sigma),$$

де k – це коефіцієнт масштабування.

3. Локалізація екстремумів.

Здійснюється пошук локальних екстремумів у просторі масштабу та просторових координатах, використовуючи 3D-матрицю, яка складається з декількох рівнів різних октав. Ключові точки визначаються як ті, що є максимальними або мінімальними відносно своїх сусідів у сусідніх масштабах і просторових координатах.

4. Оцінка розташування і фільтрація.

Екстремуми перевіряються на стабільність, використовуючи **тест Хесса**, і ті, що є нестабільними або мають низьку контрастність, відкидаються. Для аналізу також використовується обчислення матриці Гессе:

$$H = \begin{bmatrix} D_{xx} & D_{xy} \\ D_{yx} & D_{yy} \end{bmatrix}$$

5. Визначення напрямку.

Для кожної ключової точки визначається орієнтація, використовуючи градієнти інтенсивності пікселів в околі ключової точки. Це забезпечує алгоритму інваріантність до обертання. Градієнти обчислюються як:

$$m(x,y) = \sqrt{\left(I(x+1,y) - I(x-1,y)\right)^2 + \left(I(x,y+1) - I(x,y-1)\right)^2}$$

$$\theta(x,y) = tan^{-1}\left(\frac{I(x,y+1) - I(x,y-1)}{I(x+1,y) - I(x-1,y)}\right),$$

де I – інтенсивність пікселя.

6. Створення дескриптора.

В околі кожної ключової точки створюється гістограма напрямків градієнтів. Ці гістограми формуються в 128-елементні вектори, які є дескрипторами ключових точок.

Наведемо приклад коду для обробки окремого зображення за допомогою цього методу з використанням бібліотеки OpenCV на мові Python:

```
import cv2
# Створення об'єкта SIFT
sift = cv2.SIFT create()
# Виявлення ключових точок та описів
image = cv2.imread('image.jpg',
cv2.IMREAD_GRAYSCALE)
keypoints, descriptors =
sift.detectAndCompute(image, None)
# Відображення ключових точок на
зображенні
output_image = cv2.drawKeypoints(image,
keypoints, None,
flags=cv2.DRAW_MATCHES_FLAGS_DRA
W_RICH_KEYPOINTS)
cv2.imshow('SIFT Keypoints', output_image)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

Для підготовки датасету, придатного для застосування SIFT, зображення повинні бути отримані за різних умов освітлення,

кутів зйомки та масштабів. Це дозволить алгоритму виділити універсальні ознаки, стійкі до змін. Також потрібно забезпечити наявність пар зображень з відомими відповідностями, щоб навчити алгоритм коректно знаходити ключові точки на зображеннях різних масштабів та орієнтацій.

Забезпечення таких умов дозволить отримати безшовний набір даних, де виділені ознаки будуть інваріантними, що полегшить задачу виявлення відповідностей під час польоту.

Таким чином, SIFT у бібліотеці OpenCV можна використовувати для виявлення ключових точок і створення їхніх дескрипторів, що є основою для подальшого порівняння з іншими зображеннями.

Speeded-Up Robust Features (SURF)

SURF був розроблений Г. Бейєм, Т. Леманом і Л. ван Гоолом у 2006 році як покращення SIFT. Він використовує інтегральні зображення для значного прискорення процесу виділення ознак і є менш обчислювально витратним порівняно з SIFT. Алгоритм складається з кількох етапів:

<u>1. Інтегральне зображення.</u>

Інтегральне зображення – це спеціальне представлення зображення, яке дозволяє обчислювати суму пікселів у будь-якій прямокутній області за постійний час. Інтегральне зображення обчислюється як:

$$I_{integral}(x,y) = \sum_{x' \leq x, y' \leq y} I(x',y')$$

<u>2. Блочний фільтр Хаара.</u>

SURF використовує Хаар-подібні фільтри для обчислення градієнтів, але, на відміну від SIFT, виконує швидке обчислення за допомогою інтегральних зображень. Важливою ознакою даного методу є застосування блочних фільтрів Хаара для обчислення похідних другого порядку, що дозволяє оцінювати зміни в інтенсивності на різних масштабах.

<u>3. Визначення ключових точок.</u>

Ключові точки обчислюються як локальні максимуми відповіді Лапласіана, що використовується для швидшого виявлення змін.

<u>4. Створення дескриптора.</u>

SURF використовує квадратний регіон навколо ключової точки і ділить його на 16 підрегіонів. Для кожного підрегіону обчислюється сума величин градієнтів і їх напрямків, що формує вектор ознак довжиною 64 елементи.

SURF раніше був включений в бібліотеку OpenCV, однак через патентні обмеження він був вилучений з основного пакету. Водночас його можна використовувати, якщо встановити OpenCV з додатковими модулями (opencv_contrib).

Наведемо приклад коду для обробки окремого зображення за допомогою цього методу з використанням бібліотеки OpenCV на мові Python:

```
import cv2
# Створення об'єкта SURF (потрібно OpenCV з contrib)
surf = cv2.xfeatures2d.SURF_create(hessianThreshold=400)
# Виявлення ключових точок та описів
keypoints, descriptors = surf.detectAndCompute(image, None)
# Відображення ключових точок на зображенні
output_image = cv2.drawKeypoints(image, keypoints, None, flags=cv2.DRAW_MATCHES_FLAGS_DRAW_RICH_KEYPOINTS)
cv2.imshow('SURF Keypoints', output_image)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

При підготовці датасету для SURF важливо використовувати зображення різних роздільних здатностей для того, щоб SURF міг виділити ключові точки в умовах змінного масштабу, а також забезпечити наявність зображень із різноманітними текстурами, щоб алгоритм міг виділяти стійкі ознаки навіть у випадку слабко текстурованих поверхонь.

Отже SURF можна використовувати для швидкого і стійкого виділення ознак, що особливо важливо для реальних систем, де обмежені обчислювальні ресурси.

Oriented FAST and Rotated BRIEF (ORB)

ORB був розроблений у 2011 році і є комбінацією FAST (для виявлення ключових точок) і BRIEF (для створення бінарних дескрипторів). Це алгоритм, який був створений як більш швидка та відкрита альтернатива SIFT і SURF.

1. Виявлення ключових точок FAST.

FAST – це метод, який використовує простий тест для виявлення куточків. Для кожного пікселя порівнюється інтенсивність цього пікселя з інтенсивністю пікселів, розташованих на колі радіуса 3. Піксель вважається ключовою точкою, якщо група з мінімум 12 сусідів має інтенсивність, значно вищу або нижчу за інтенсивність центрального пікселя.

2. Призначення орієнтації.

Після виявлення ключових точок, для кожної з них обчислюється напрямок за допомогою методу моментів, що дозволяє обчислити напрямок градієнта і зробити ORB інваріантним до обертання.

3. Створення дескриптора BRIEF.

BRIEF створює бінарний дескриптор, використовуючи порівняння інтенсивностей у випадкових парах пікселів в околі ключової точки. Це дозволяє швидко обчислювати подібність між дескрипторами під час пошуку відповідних точок між зображеннями.

Для кожної пари пікселів (p, q) бінарне значення визначається як:

$$f(p,q) = \begin{cases} 1,\text{якщо } I(p) < I(q) \\ 0,\text{інакше} \end{cases}$$

4. Орієнтований BRIEF.

ORB використовує модифіковану версію BRIEF, яка враховує обертання. Під час обчислення дескрипторів окремі пари точок обертаються на кут, відповідний орієнтації ключової точки.

ORB реалізований в OpenCV як вбудована функція.

Наведемо приклад коду для обробки окремого зображення за допомогою цього методу з використанням бібліотеки OpenCV на мові Python:

```
import cv2
# Створення об'єкта ORB
orb = cv2.ORB_create()
# Виявлення ключових точок та описів
keypoints, descriptors = orb.detectAndCompute(image, None)
# Відображення ключових точок на зображенні
output_image = cv2.drawKeypoints(image, keypoints, None, flags=cv2.DRAW_MATCHES_FLAGS_DRAW_RICH_KEYPOINTS)
cv2.imshow('ORB Keypoints', output_image)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

При підготовці датасету для методу ORB необхідно забезпечити наявність зображень, отриманих з різних кутів, щоб забезпечити стійкість до змін орієнтації. Також необхідно, щоб зображення були точно зареєстровані, щоб ORB міг використовувати їх для навчання на парах зображень із відомими відповідностями.

Таким чином ORB забезпечує швидке виділення ознак і може використовуватися у реальному часі на менш потужних пристроях, таких як дрони, де обмежені обчислювальні ресурси.

Експериментальне порівняння методів

Для визначення можливостей використання методів виділення ознак SIFT, SURF, ORB для навчання системи, здатної коректно порівнювати зображення в режимі реального часу, нами було проведено експериментальне дослідження на більш ніж 200 тисячах тайтлів супутникових зйомок Bing Maps з розмірами 256 на 256 пікселів та зумом 17. Для проведення експерименту було розроблено програму на мові програмування C++ з використанням бібліотеки OpenCV та QT Widgets, що працювала у багатопоточному режимі для пришвидшення розрахунків з використанням CPU [6]. Кожне зображення оброблялося трьома різними методами (SIFT, SURF, ORB) у рамках одного потоку та здійснювався запис швидкості виявлення

ознак кожним методом. Результати відображено на діаграмі (рис. 1).

Експеримент показав, що найшвидшим методом є метод ORB, а найповільнішим є метод SIFT. Різниця швидкості обробки корелюється з середньою кількістю ознак (рис. 2), однак метод ORB є приблизно удвічі швидшим по абсолютній швидкості ніж два інші методи.

Унаслідок експерименту встановлено, що метод SURF є суттєво ефективнішим за інші протестовані методи при роботі з зображеннями місцевості з малою кількістю унікальних об’єктів (рис. 3 та 4).

Також було проаналізовано доцільність паралельного використання різних методів. Було виявлено, що більшість зображень, у яких було виявлено недостатню кількість ключових точок, або ключових точок не було виявлено взагалі, є спільними для всіх перевірених методів, однак метод SURF дає мінімальну кількість унікальних похибок (рис. 5 та 6).

З метою забезпечення прив’язки місцезнаходження БПЛА зі збігом сцени з використанням методів ШІ, необхідно забезпечити навчання системи.

Навчання системи розпочинається з того, що обраний датасет піддається попередній обробці для виділення ключових точок, які є інваріантними до змін масштабу, освітлення і обертання. Ключові точки, знайдені за допомогою SIFT, SURF або ORB, у подальшому використовуються як основа для створення дескрипторів. Дескриптори є математичним представленням особливостей зображення і служать для порівняння між різними кадрами.

Під час навчання застосовуються методи машинного навчання, які навчаються на виявлених ознаках, щоб розпізнавати відповідності між кадрами у різних умовах. Наприклад, навчальні алгоритми можуть використовувати техніки кластеризації, такі як k-means, для створення візуальних слів (visual words), що дозволяє зменшити розмірність простору дескрипторів і спростити задачу класифікації ознак. Це дозволяє створити ефективну модель для порівняння ознак між зображеннями під час польоту.

Після навчання система отримує можливість використовувати отримані моделі для обробки нових кадрів, які надходять з камери під час польоту БПЛА. Нові зображення також обробляються для виділення ключових точок, і отримані дескриптори порівнюються з тими, що були збережені у ході навчання. Відповідність між ознаками дозволяє оцінити геометричні перетворення та забезпечити позиціонування БПЛА відносно зображень, отриманих під час попередньої підготовки місії.

Для навчання моделей з використанням ознак, отриманих через SIFT, SURF або ORB, необхідно створити дескриптори для всіх зображень та використовувати їх для подальшого навчання за допомогою алгоритмів машинного навчання, таких як Random Forest або Support Vector Machines (SVM). Наприклад, для кожного зображення у датасеті можна обчислити дескриптори за допомогою SIFT, а потім зберегти ці дескриптори як вихідні дані для моделі.

Для реалізації навчання моделі необхідно використати вектори ознак як вихідні дані для навчання моделей з застосуванням методів Random Forest, SVM. Так SVM може бути використаний для класифікації відповідностей між зображеннями, а Random Forest - для швидкої класифікації у реальних умовах. Цей підхід дозволяє використовувати попередньо обчислені ознаки для навчання моделей, що може бути особливо корисно, коли доступні ресурси обмежені, а також якщо потрібна швидка обробка у реальному часі.

Існує також альтернативний підхід, який передбачає використання ковзного вікна (sliding window), при якому отриманий кадр з камери порівнюється з масштабованим до потрібного масштабу кадром із датасету. Цей метод дозволяє знайти відповідність між зображеннями без попереднього навчання моделі, але має свої обмеження. Зокрема, точність цього підходу значно залежить від роздільної здатності та умов зйомки, а також від того, наскільки добре масштабування підходить до реальних умов польоту і не є

інваріантною до різних умов, таких як освітлення або ракурси.

Для навчання на підготовленому датасеті можна використовувати готові моделі, зокрема Bag of Visual Words (BoVW), Random Forest, Convolutional Neural Networks (CNNs), або моделі з популярних бібліотек (TensorFlow або PyTorch тощо), залежно від складності задачі та обсягу даних. Модель BoVW використовується для створення візуальних слів із дескрипторів ознак, що дозволяє порівнювати зображення на основі знайдених кластерів. Random Forest може використовуватися для класифікації відповідностей, особливо якщо потрібно швидке рішення для реальних умов. CNNs забезпечують високу точність і глибоку обробку, але потребують більше обчислювальних ресурсів.

Модель MobileNetV2 або інші моделі глибокого навчання, такі як InceptionV3, можна використовувати як варіанти перенавчання для підготовленого датасету. Однак їхнє використання вимагає перевірки та тестування, як і пошук найбільш оптимального рішення для визначення координат методом ознак.

Висновки

Питання забезпечення керування БПЛА та визначення їх місцезнаходження **є** актуальним для їх використання як у цивільних, так і військових цілях. Для запобігання впливу засобів РЕБ необхідно забезпечити можливість вирішення цього завдання без використання системи супутникової навігації. Використання прив'язки місцезнаходження БПЛА за збігом сцени є одним з можливих варіантів вирішення цієї проблеми. Вказане завдання може бути вирішено із застосуванням стандартних методів з бібліотек OpenCV, scikit-learn, tensor flow. За результатами експерименту, найбільш ефективним методом визначення ознак на супутникових знімках виявився SURF. При цьому паралельне використання двох чи трьох методів не призводить до суттєвого підвищення визначення ознак.

У подальшому нами буде проаналізовано різні алгоритми визначення місцезнаходження БПЛА із використанням визначених ознак та досліджено їх ефективність для задач визначення місцезнаходження у режимі реального часу.

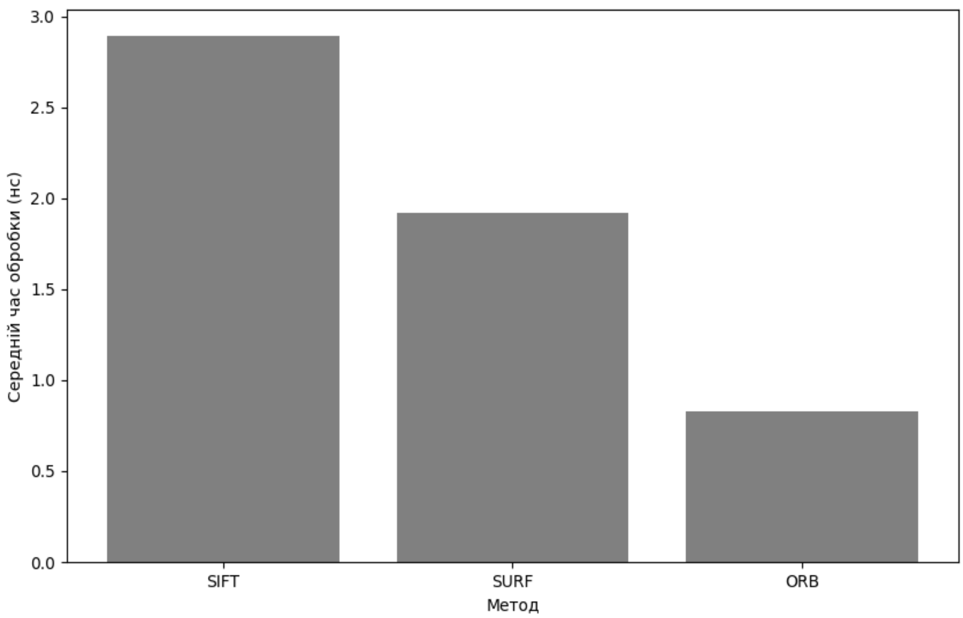


Рис. 1. Середній час обробки зображень при використанні методів SIFT, SURF, ORB

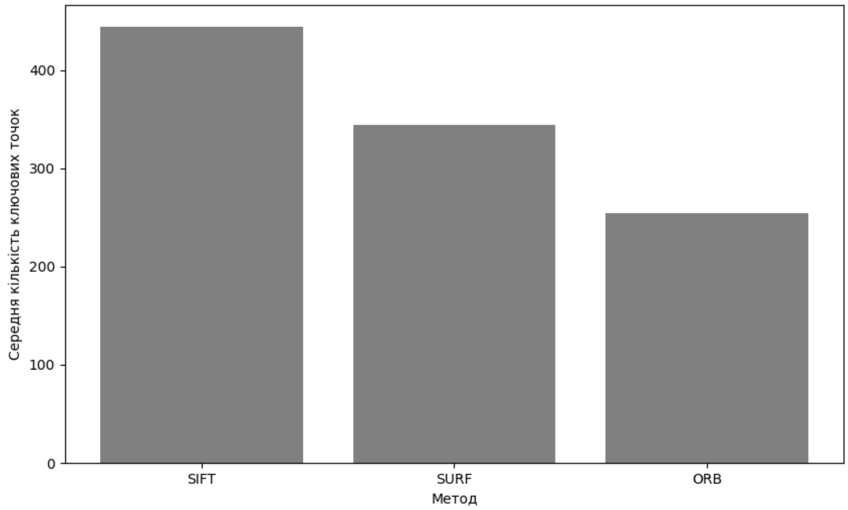


Рис. 2. Середня кількість ключових точок при використанні методів SIFT, SURF, ORB

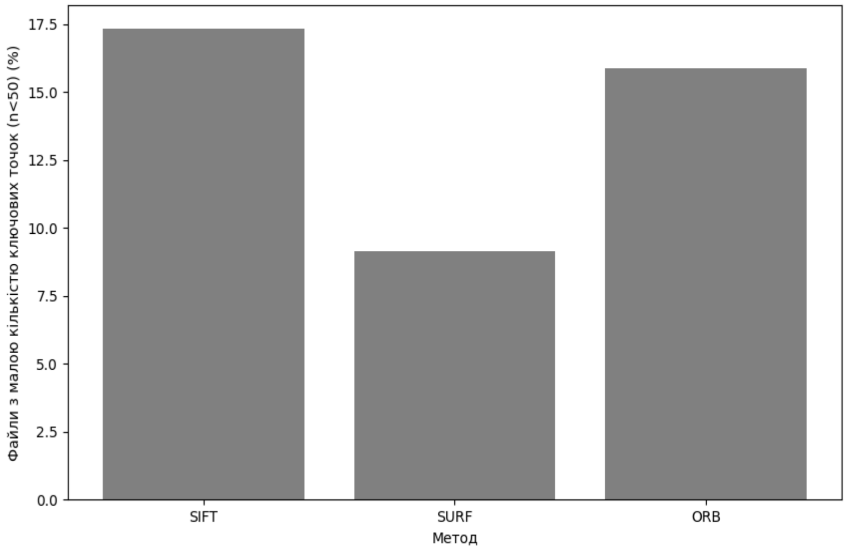


Рис. 3. Відсоток файлів з малою кількістю ключових точок при використанні методів SIFT, SURF, ORB

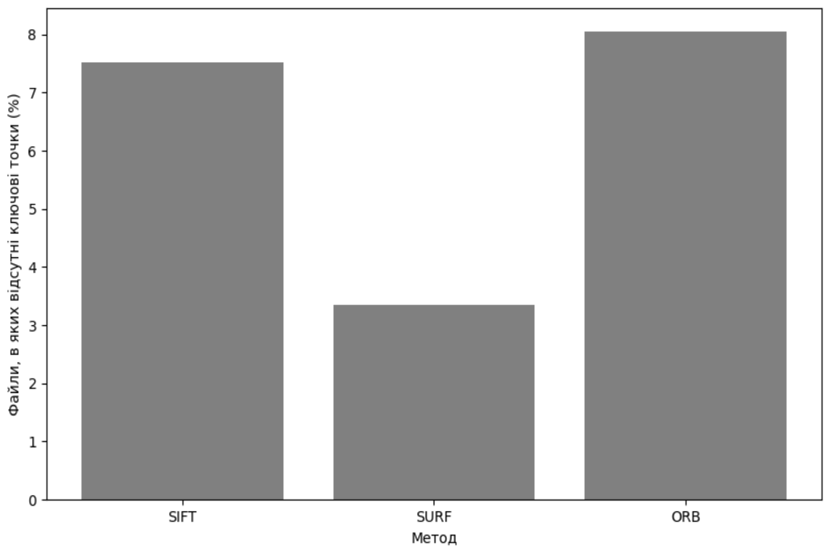


Рис. 4. Відсоток файлів, в яких відсутні ключові точки, при використанні методів SIFT, SURF, ORB

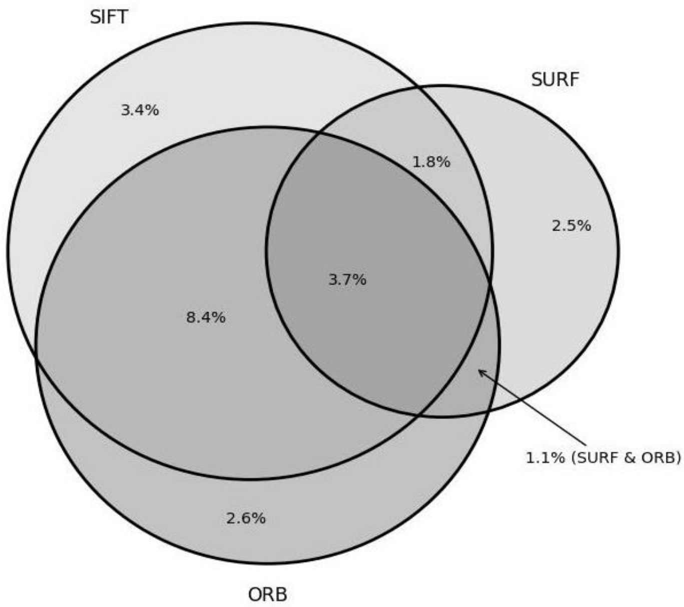


Рис. 5. Файли з малою кількістю ключових точок при використанні методів SIFT, SURF, ORB

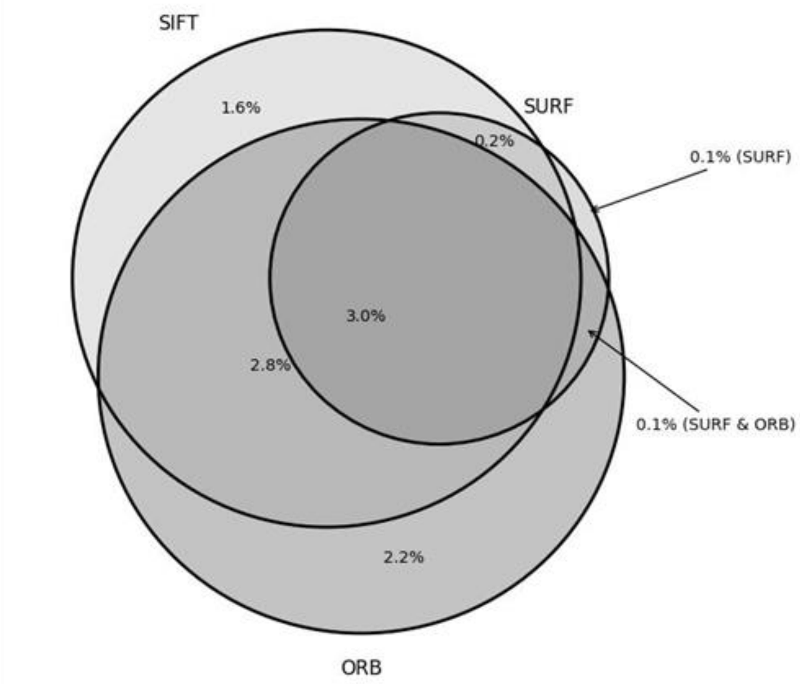


Рис. 6. Файли, в яких відсутні ключові точки, при використанні методів SIFT, SURF, ORB

Література

1. Wrigley W. History of Inertial Navigation. *Navigation*. Vol. 24. Issue 1. Режим доступу: URL https://doi.org/10.1002/j.2161-4296.1977.tb01262.x (звернення 01.10.2024)

2. Riedel F. W., Hall S. M., Barton J. D. та ін. “Guidance and Navigation in the Global Engagement Department,” *Johns Hopkins APL Technical Digest.* 2010. 29 (2). P. 118–132.

3. Lowe D. G. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision.* 2004. 60 (2). P. 91–110.

4. Bay H., Tuytelaars T. & Van Gool, L. SURF: Speeded up robust features. *European Conference on Computer Vision.* Springer, 2006. P. 404–417.

5. Rublee E., Rabaud V., Konolige K., & Bradski G. ORB: An efficient alternative to SIFT or SURF. *International Conference on Computer Vision*. IEEE, 2011. P. 2564–2571.

6. Лукацький Є. Вихідний код програми дослідження методів виділення ознак SIFT, SURF, ORB. Режим доступу: https://github.com/Zhekar1998/Article_scene_matching_app.git

References

1. Wrigley, W. History of Inertial Navigation. *Navigation* vol 24, issue 1 Available: URL:https://doi.org/10.1002/j.2161-4296.1977.tb01262.x.

2. Riedel, F. W., Hall, S. M., Barton, J. D., та ін. (2010). Guidance and Navigation in the Global Engagement Department, Johns Hopkins APL Technical Digest, 29(2), 118–132.

3. Lowe, D. G. (2004). Distinctive image features from scale-invariant keypoints. International Journal of Computer Vision, 60(2), 91–110.

4. Bay, H., Tuytelaars, T., & Van Gool, L. (2006). SURF: Speeded up robust features. In European Conference on Computer Vision (pp. 404–417). Springer.

5. Rublee, E., Rabaud, V., Konolige, K., & Bradski, G. (2011). ORB: An efficient alternative to SIFT or SURF. In 2011 International Conference on Computer Vision (pp. 2564–2571). IEEE.

6. Lukatskyi, Ye. Vykhidnyi kod prohramy doslidzhennia metodiv vydilennia oznak SIFT, SURF, ORB. Available: https://github.com/Zhekar1998/Article_scene_matching_app.git.

The article has been sent to the editors 15.11.24.
After processing 18.11.24.
Submitted for printing 30.12.24.